NOMBRE: ___*CLAVE*_____ **SECCION** _____

¡Anota tu nombre completo en esta página y tus iniciales en <u>todas las demás hojas</u> del examen AHORA! (penalidad de 5 puntos)

Tienes 1.5 horas para completar todos los problemas. Lee cuidadosamente todo el examen antes de empezar a trabajar. Muestra todo el trabajo conducente a tu contestación. Podrás recibir crédito parcial por contestaciones parciales siempre y cuando muestres tu trabajo por escrito. Usa tu tiempo inteligentemente. Exito!

ICOM 4036 Staff

1	25
2	35
3	20
4	10
5	10
Total	100

Problem 1. (25 points) Low-level Expression Evaluation

1. **(5 pts each) Match each of the algebraic expressions in the left column with ZERO OR MORE corresponding equivalent sequences of Easy I assembly language code segments in the right column. Assume that a segment that works must leave the correct value of the expression in the accumulator (Easy I). Variables are stored in an address labeled by their name.**

Pseudo-Instruction	Native Sequence
_a_ x – (x + 1)	a. loadi X addi 1 comp addi 1 add X
_e_ x + x - 1	b. loadi X comp addi 1 add X addi 1
_d_ x - x - 2	c. loadi X add X addi 1 comp addi 1
_b_ (x – x) + 1	d. loadi X addi 1 addi 1 comp addi 1 add X
___ (x – 1) * 2	e. andi 0 addi 1 comp addi 1 add X add X

Problem 2. (35 points) High-level programming language implementation

Consider the following segment of C code:

```
int a[5] = { 3, 1, 4, 5, 2};
int max=0;
for (int i=0; i<5; i++) {
     if (a[i] > max) max = a[i];
}
```

(a) (5 points) What will be the value of variable *max* at the end of the segment

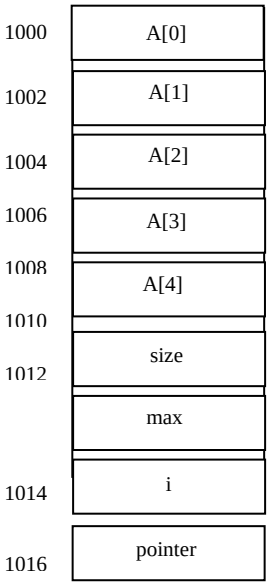
5

(b) (15 points) Provide an equivalent program segment in MIPS assembly language

```
            .data                   # int a[5] = { 3, 1, 4, 5, 2};
A           .word       3           #
            .word       1           #
            .word       4           #
            .word       5           #
            .word       2           #
size        .word       5           #
max         .word       0           #
            .global     main        #
            .text                   #
main:       la   $s4, A             #
            lw   $s0, 0($t0)        #
            la   $t1, size          #
            lw   $s1, 0($t1)        # // size in $s1
            lw   $s2, 4($t1)        # int max = 0; // max in $s2
            li   $s3, 0             # // i in $s3
for:        bge  $s3, $s1, endfor   # for (int i=0; i<5; i++) {
            lw   $s0, 0($s4)        #
if:         bge  $s2, $s0, endif    #    if (a[i] > max)
            move $s2, $s0           #        max = a[i];
endif:      addi $s4, 4             #
            addi $s3, 1             #
            j    for                # }
endfor:                             #
```

(c) (15 points) Provide an equivalent program segment in Easy I assembly language. You must assign memory locations for each of the variables.

```
        andi        0
        addi        3
        storei      1000
        andi        0
        addi        1
        storei      1002
        andi        0
        addi        4
        storei      1004
        andi        0
        addi        5
        storei      1006
        andi        0
        addi        2
        storei      1008
        andi        0
        addi        5
        storei      1010
        andi        0
        addi        5
        storei      1010
        andi        0
        addi        0
        storei      1012
        andi        0
        addi        0
        storei      1014
for:    loadi       1010
        comp
        addi        1
        add         1014
        add         -1
        brni        endfor
        loadi       1014
        add         1014
        addi        1000
        storei      1016
        load        1016
        comp
        addi        1
        add         1012
        brni        endif
        load        1016
        storei      1012
endif:  loadi       1014
        addi        1
        storei      1014
        jumpi       for
endfor:
```



Problema 3. (20 points) TRUE of FALSE Potpurri

(2 points each) Indicate which of the following statements is True (T) or False (F) by enclosing the answer in a circle.

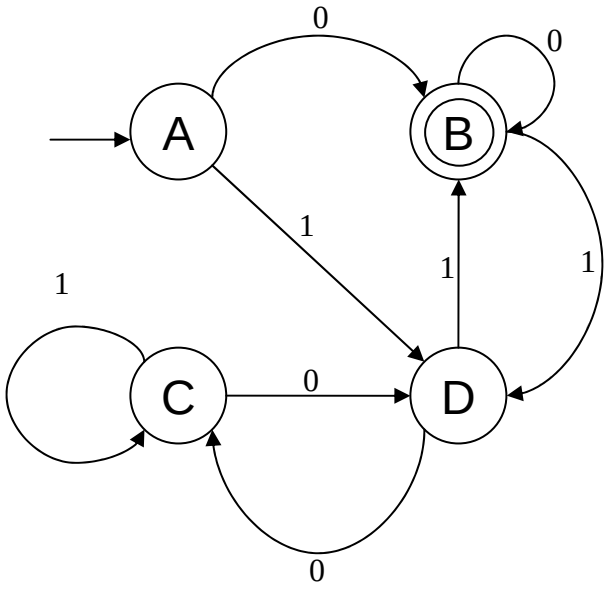
Some of the statements will make reference to the following grammar G:

```
S -> S S
   | (S)
   | ϵ
```

1. **(T or F) G is ambiguous**
2. **(T or F) G can be parsed by a recursive descent parser**
3. **(T or F) The string `((()())())` belongs to the language generated by G**
4. **(T or F) A string generated by G must contain at least one balanced pair of parenthesis**
5. **(T or F) A language that forbids infinite recursions is more powerful than a Turing Machine**
6. **(T or F) A Turing machine has infinite memory**
7. **(T or F) The are many problems that are not computable by Turing Machines**
8. **(T or F) General purpose processors are typically interpreters of some machine code language**
9. **(T or F) Low-level languages are heavily dependant on processor architecture**
10. **(T or F) Part of the job of the parser is to determine is variables are defined before they are used**

Problema 4. (10 points) Finite State Machines

Complete the transitions to make the finite state diagram below recognize the language of binary numbers of any length that are divisible by 3. The initial state is A and the alphabet consists of {0,1}.



Problem 5. (10 points) Course Evaluation

Es importante que completes esta parte con la mayor seriedad e interés posibles. Tu contribución ayudará a mejorar la calidad del curso significativamente.

Gracias.

1) Menciona los tres aspectos que mas te gustan de la clase ICOM 4036

a)

b)

c)

2) Menciona los tres aspectos que menos te gustan de la clase ICOM 4036

a)

b)

c)